

Spectral Methods for Matrices and Tensors

Ravindran Kannan
Microsoft Research Labs., India. ^{*†}

April 9, 2010

Abstract

While Spectral Methods have long been used for Principal Component Analysis, this survey focusses on work over the last 15 years with three salient features: (i) Spectral methods are useful not only for numerical problems, but also discrete optimization problems (Constraint Optimization Problems - CSP's) like the max. cut problem and similar mathematical considerations underlie both areas. (ii) Spectral methods can be extended to tensors. The theory and algorithms for tensors are not as simple/clean as for matrices, but the survey describes methods for low-rank approximation which extend to tensors. These tensor approximations help us solve Max- r -CSP's for $r > 2$ as well as numerical tensor problems. (iii) Sampling on the fly plays a prominent role in these methods. A primary result is that for any matrix, a random submatrix of rows/columns picked with probabilities proportional to the squared lengths (of rows/columns), yields estimates of the singular values as well as an approximation to the whole matrix.

1 Introduction

Spectral methods have been widely used in many areas for Numerical problems under the name Principal Component Analysis. The algorithmic use

^{*}kannan@microsoft.com

[†]©This is the author's personal version of the work. It is posted here by permission of ACM for personal use, not for distribution. The definitive version is published in the Proceedings of the ACM Symposium on Theory of Computing, 2010.

of spectral methods for discrete problems is perhaps more recent. The first point of this survey is to suggest that similar mathematical considerations motivate both discrete and numerical applications.

Our second point is the extension of algorithms to tensors. Linear Algebra is unique in that it has beautiful theory which also translates to efficient as well as optimal algorithms. Tensors admit no such comparable theory or algorithms; indeed, some impossibility or hardness results are known for tensors. This is not our focus. Instead, we want to see what is algorithmically possible with tensors, of course, having to be less ambitious than for matrices. [But we seek provable error bounds.] The second purpose of this survey, then, is to present methods for matrices which extend to naturally to tensors. Two methods surveyed here - the Cut Norm introduced in section 2 and Length-Squared sampling procedure introduced in section 5 both have this flavor.

The third main point of the survey is faster randomized algorithms for large matrices (based on sampling) than traditional numerical algorithms; here length-squared sampling provides a starting point. Besides improving running time through sampling, one operates in a model where the massive matrices cannot be stored in Random Access Memory, but must be read and sampled on the fly. Several newer sampling procedures are also covered in section 7. Many open questions remain both in regard to algorithms and computational applications. We will list them in the text, but, here we mention two generic challenges to highlight possible directions for research.

Challenge 1: Find the spectrum of the web graph. The web graph (of hypertext links, for example) is a large “naturally occurring” graph. It is directed and so the adjacency matrix is not symmetric. The question is to find the singular values approximately. We will see that the randomized algorithms in the survey could be useful for this. But the provable upper bounds on sample size are too large. Can smaller sample sizes suffice? We want a certificate on the error bound and confidence (probability of correctness) of the spectrum (say at least of the largest several hundred singular values) as computed for the particular matrix. [In a sense, this is seeking to make a Monte-Carlo algorithm into a Las Vegas one.] The motivation for the particular question is: there is much research on statistical properties of the web graph. But most, pertain to “local properties” - degree distribution (which is local in that we find the degree of one vertex at a time), local communities (in a neighborhood of one vertex), etc. The top singular value is a measure of “global” correlation. Further, the important notion of pagerank [BP98] as well as Kleinberg’s HITS algorithm [Kle99] of course have links to

the spectrum of the graph.

Challenge II Better provable Algorithms for Tensors

The second challenge is: while the methods here make a beginning in dealing with tensors much research remains to be done on algorithms for tensors. We seek better methods for maximizing (approximately) cubic and higher order forms (especially when there are unusually good solutions) and finding low-rank approximations to tensors faster.

Some notation For a matrix A , the Frobenius norm $\|A\|_F$ is the square root of the sum of the squares of its entries. The spectral norm $\|A\|_2$ is $\max_{x:|x|=1} |Ax|$. The abbreviation u.a.r stands for “uniformly at random”. ϵ will be a positive error parameter. The material in sections 4 onwards (which may be read more or less independently of the earlier sections) is covered in greater detail in the monograph [KV08]. Spectral Graph Partitioning (starting with Fiedler’s work [Fie75]) and many other important topics which are well covered elsewhere are not dealt with here.

2 Approximation of matrices in Cut-norm, applications

Instead of the traditional low-rank approximations to matrices, we start with a form more suited to discrete applications. For an $m \times n$ matrix, a “rectangle” will mean one of the 2^{m+n} sets of the form (a subset of rows) \times (a subset of columns). A “cut matrix” is an $m \times n$ matrix with all its entries in some rectangle equal and all entries outside this rectangle being zero; it is special rank 1 matrix. Our approximation to a matrix is by a sum of cut matrices. These approximations have the following desirable properties:

1. It is very easy to show they exist.
2. They can be found (and this in non-trivial) in polynomial, in fact in constant time (implicitly) with uniform random sample of $O(1)$ entries from the matrix.
3. Using this, one can solve the maximum cut problem to additive error $\epsilon n^2 M$ on n node graphs with maximum edge weight M . [This also extends to all MAX-2-CSP’s.]

4. Both the existence as well as the algorithmic version (in constant time) can be extended to tensors and using this, one can approximately solve MAX-r-SAT for fixed r and other problems.

This is all described in this section. The main caveat for these methods is the error of $\epsilon n^2 M$. There are many applications where this is not good enough. In the discrete optimization setting (say as in Max Cut), if $M = 1$, the max cut needs to be $\Omega(n^2)$ or equivalently, the graph needs to be dense for this error to be useful. Of course, many interesting problems are not dense. A similar situation holds for PCA as well. Section 4 describes a more amenable error bound for both these areas and how to achieve it using non-uniform sampling and thus address application issues.

A, B will stand for $m \times n$ matrices. If S is a subset of rows and T a subset of columns, then, we let

$$A(S, T) = \sum_{i \in S, j \in T} A_{ij}.$$

Define the “cut norm” of $\|A\|_{\square}$ by

$$\|A\|_{\square} = \max_{S, T} |A(S, T)|.$$

Lemma 1. *Assume $|A_{ij}| \leq 1$. There exist $1/\epsilon^2$ cut matrices whose sum B approximates A in the sense*

$$\|A - B\|_{\square} \leq \epsilon mn.$$

Proof: If $\|A\|_{\square} \leq \epsilon mn$, we can take $B = 0$. Otherwise, there is some S, T such that $|A(S, T)| \geq \epsilon mn$. Our first cut matrix in B has $A(S, T)/|S||T|$ in each entry of $S \times T$ and zero elsewhere. This subtracts from each $A_{ij}, i \in S, j \in T$, their average and it is easy to show that then $\sum_{i \in S, j \in T} A_{ij}^2$ decreases by at least $\epsilon^2 mn$ and so does $\|A\|_F^2$. Since $\|\cdot\|_F \geq 0$, the process must terminate in at most $1/\epsilon^2$ steps proving the Lemma.

An immediate question is whether such a B can be found in polynomial time. An affirmative answer was given by Frieze and Kannan [FK99] and indeed, they proved that it can be found in “constant time” in a sense to made clear later.

Theorem 1. [FK99] *For A with $|A_{ij}| \leq 1$ and any fixed $\epsilon > 0$, we can find in polynomial time a matrix B which is the sum of $4/\epsilon^2$ cut matrices and*

satisfies $\|A - B\|_{\square} \leq \epsilon mn$. In fact, given the entries of A in just a u.a.r. rectangle of size $O^*(1/\epsilon^4) \times O^*(1/\epsilon^4)$, we can find an implicit description of B .

From the proof of the Lemma (1), we see that it suffices to determine if the cut norm of A is at most ϵmn and if not, find an S, T with $|A(S, T)| \geq \epsilon mn$. The exact problem is NP-hard. However, for the theorem an approximate version suffices: find the maximum value of $|A(S, T)|$ to within additive error ϵmn . [Really $\frac{\epsilon}{2}mn$, but redefine ϵ to avoid putting $/2$ etc.] Reduces to two problems: $\text{Max } A(S, T)$ and $\text{Max } -A(S, T)$. We describe the ideas behind a polynomial time algorithm for $\text{Max} A(S, T)$ which we may call

The Maximum Rectangle Problem:

- **S gives T**

If we know the maximizing S , the T to go with it is just the columns of A whose sum in the S rows is positive.

- **Estimate Column sums in S rows**

Pick a subset W of $s = O(1/\epsilon^2)$ rows u.a.r. The sum of each column in the S rows can be estimated (to additive error ϵmn) by $\frac{m}{s} \times$ sum in the $W \cap S$ rows.

- **Exhaustive Enumeration**

Don't know S or $S \cap W$. But, we can try each subset \tilde{W} of W (there are only 2^s of them) as a candidate $S \cap W$. For each, find the set of columns- T - whose sum in the \tilde{W} rows is positive.

- **Choose best candidate**

For each candidate T : Let S' be the rows with positive sum in the T columns. Take $\max A(S', T)$ among all candidate T .

The Exhaustive enumeration step is inspired by an idea of Arora, Karger and Karpinski [AKK95]. Why does this algorithm work? We supply only a brief intuition. In the estimating column sums step, if we had the correct $S \cap W$, the only columns on which we could be wrong about the sign of the column sum in the S rows are ones where the column sum is close to 0. But these do not contribute much to $A(S, T)$ anyway. So, one of our candidate T in step 4 is correct in the sense that $A(S, T)$ is high for the true S . The

last step finds the best T among the candidates; we didn't need the true (unknown) S ; the best S for each T is just the S' .

How do we make this all constant time ? The idea is simple : Pick a u.a.r subset \hat{S} of $\hat{s} = O^*(1/\epsilon^4)$ rows and a u.a.r. subset \hat{T} of \hat{s} columns at the outset. In the above algorithm, instead of finding $A(S', T)$ for each of the $2^{(1/\epsilon^2)}$ candidate T 's, estimate this quantity by $\frac{mn}{\hat{s}^2} A(S' \cap \hat{S}, T \cap \hat{T})$, for which we only need to know the entries of A in $\hat{S} \times \hat{T}$. One can show using Höfdding-Azuma inequality that the estimate is within additive error at most $O(\epsilon mn)$ with high probability. [The failure probability is at most $e^{-\hat{s}/\epsilon^2} = e^{-1/\epsilon^2}$ for each candidate; so by the union bound, whp, there is no failure for any candidate.] Hence the best candidate is found with asserted error whp.

The approximation B has many algorithmic uses. First consider the Maximum cut problem in an undirected graph. The problem can obviously be written as (with $A =$ the $n \times n$ adjacency matrix of the graph)

$$\text{MAX}_{x \in \{0,1\}^n} x^T A(1 - x),$$

where x is a column vector and 1 is the vector of all 1's. We obviously have

$$|x^T A(1 - x) - x^T B(1 - x)| \leq \|A - B\|_{\square}.$$

Here, we see why the cut norm is defined the way it is; it is the most natural norm for ensuring that for 0-1 vectors x, y , $x^T A y \approx x^T B y$. [It is close to a more traditional "operator norm" - namely, it is easy to show that $\|A\|_{\square}$ is always within a factor of 4 of $\text{MAX}_{|x|_{\infty}=|y|_{\infty}=1} x^T A y$.] So to get the maximum cut within additive error ϵn^2 , it suffices to solve

$$\text{MAX}_{x \in \{0,1\}^n} x^T B(1 - x).$$

Since B has constant (depending only on ϵ , not on n) rank, $x^T B(1 - x)$ is determined by a constant number of variables, namely the components of x along the space spanned rows/columns of B . Thus, we have reduced the n variable problem to one with a constant number of variables. While there are many technical difficulties in solving the problem with B , conceptually, it is simple to argue that it can be solved in time exponential in the rank of B alone by enumeration: Put a fine enough grid in the row/column space of B . The number of grid points is exponential only in the dimension of the space. For each grid point, the value of $x^T B(1 - x)$ is determined. It only

remains to know which grid points correspond to 0-1 x 's. This is an integer program; but its relaxation Linear Program turns out to suffice for the error we seek.

Indeed, the attractive feature of this line is that these arguments can be extended in a straight forward manner to solve all dense MAX-2-CSP problems in polynomial time in a unified manner. Moreover, this also extends to approximating tensors in cut norm and that helps us solve all dense MAX- r -CSP problems for any fixed arity r . [Recall: In a MAX- r -CSP problem, one is given a list of m Boolean functions - f_1, f_2, \dots, f_m , each a function of only r of the variables. We have to find a truth setting of all variables which satisfies as many of the m functions as possible. MAX- r -SAT where each function is the disjunction of r literals is a central example.] Indeed, one gets:

Theorem 2. [FK99][AE02] *Any MAX- r -CSP problem on n variables, where r is fixed, can be solved to additive error ϵn^r in constant time for fixed $\epsilon > 0$. [The running time depends exponentially on $O^*(1/\epsilon^2)$.]*

For $r = 2$, the area of Property Testing [GGR98] proved the first such results by combinatorial means, often by exploiting the structure of particular problems. A flavor of the combinatorial difficulties is seen from the first problem so attacked - max.cut. by DelaVega [dlV96]: Akin to the Estimating columns sums and Exhaustive Enumeration steps of our algorithm for maximizing $A(S, T)$, the property testing algorithms do the following for max.cut: clearly in a max cut (S, \bar{S}) , every vertex in S has more edges to \bar{S} than to S . If we just picked a u.a.r. subset W , then try out all possible subsets \tilde{W} of W as candidate $S \cap W$, we could classify each vertex by whether it has higher degree into \tilde{W} or $W \setminus \tilde{W}$ and hope these would be respectively \bar{S} and S vertices. But this does not work and indeed, the property testing based max cut algorithms work hard to fix this. In our setting, S, T are subsets of different sets and so are “decoupled” and this is what makes these steps work so simply here.

For general $r > 2$, Andersson and Engebretsen [AE02] have given a purely combinatorial approach independently to prove theorem (2) too. Our approach to proving the theorem is based on an extension of cut matrices and norm to tensors which we describe in the next section.

Another application of the cut norm and approximation is to a version of the Szémeredi Regularity Lemma for graphs. For a graph $G(V, E)$ with edge weights, we denote by A_G the (weighted) adjacency matrix. For two graphs,

G, G' , on the same set of n vertices, we define a distance between them by $d_{\square}(G, G') = \frac{1}{n^2} \max_{S, T \subseteq V} |A_G(S, T) - A_{G'}(S, T)|$.

From Lemma 1, the following Lemma which is often called the Weak Regularity Lemma can be proved:

Lemma 2. [FK99] *The vertex set of a graph $G(V, E)$ with all edge weights equal to 1 can be partitioned into $2^{O(1/\epsilon^2)}$ subsets V_1, V_2, \dots so that the graph G' in which for each edge (i, j) , with say, $i \in V_r, j \in V_s$, has weight = (number of edges between V_r and V_s)/ $|V_r||V_s|$ has $d_{\square}(G, G') \leq \epsilon$*

Note that G' intuitively behaves like a random graph with edge probabilities given by edge weights in that the number of edges between subsets of vertices would be close to the expected numbers. A constructive version of the Szémeredi Regularity Lemma was shown in [ARH⁺]. A simpler spectral algorithm is developed in [FK999].

There is an abstract (and improved) version of this lemma due to Tulsiani, Trevisan and Vadhan [TTV09] from which they are not only able to derive this result, but several others, like the Dense Model theorem of Green, Tao and Ziegler [GT08]. A recent result of Bansal and Williams [BW09] makes progress on the classical problem of complexity of Boolean Matrix multiplication; they use Lemma 2. We will describe in the next section an application of weak regularity to graph limits.

3 Approximation of Tensors in Cut norm and applications

Recall that a r -tensor is an r dimensional array A with entries $A_{ijkl\dots}$. A “rectangle” is a set of entries of the form $S_1 \times S_2 \times \dots \times S_r$, where S_t is a subset of the t th index. [For $r = 2$, S_1 is a subset of rows and S_2 a subset of columns.] $A(S_1, S_2, \dots, S_r)$ is the sum of A ’s entries in the rectangle $S_1 \times S_2 \times \dots \times S_r$. We define the cut norm $\|A\|_{\square}$ exactly as for matrices - it is the maximum over all rectangles of $|A(S_1, S_2, \dots, S_r)|$. A cut tensor has the same entry in some rectangle and is zero elsewhere. Lemma 1 carries over with exactly the same simple proof as for matrices.

Lemma 3. [FK99] *For a r -tensor A with $n_1 \times n_2 \times \dots \times n_r$ entries, each at most 1 in absolute value, there exist $1/\epsilon^2$ cut tensors whose sum B approximates A in the sense*

$$\|A - B\|_{\square} \leq \epsilon n_1 n_2 \dots n_r.$$

Interestingly, the constructive version also carries over with one extra twist. The idea for solving the maximum rectangle problem is:

- Want S_1, S_2, \dots, S_r so that $A(S_1, S_2, \dots, S_r)$ is max.
- If we knew the maximizing S_1, S_2, \dots, S_{r-1} , then the maximizing S_r consists of the i with $A(S_1, S_2, \dots, S_{r-1}, i) > 0$.
- We can estimate this by taking random subsets W_1, W_2, \dots, W_{r-1} , trying out all subsets $\tilde{W}_1, \tilde{W}_2, \dots, \tilde{W}_{r-1}$ of the respective W_t as candidate $S_t \cap W_t$.
- This gives us many candidate S_r . How do we find the best one? This needs the extra twist: define a $r - 1$ tensor \tilde{A} by

$$\tilde{A}_{i_1 i_2 \dots i_{r-1}} = \sum_{i \in S_r} A_{i_1 i_2 \dots i_{r-1} i}.$$

Now recursively solve the maximum rectangle problem for the $r - 1$ tensor. Then choose the S_r with best answer.

The above arguments can be used to show: for a MAX-r-CSP formula $F(x_1, x_2, \dots, x_n)$ if we pick a u.a.r. subset Q with $|Q| = q = \text{poly}(1/\epsilon)$ variables and solve the “induced” MAX-r-CSP problem F^Q on the picked variables (the induced problem contains only those clauses all of whose literals are the picked variables or their negations), then we have whp:

$$\left| \frac{n^r}{q^r} \text{MAX}(F^Q) - \text{MAX}(F) \right| \leq \epsilon n^r,$$

where $\text{Max}(F)$ denotes the maximum number of functions in F which can be simultaneously satisfied. $[\frac{n^r}{q^r}]$ is a natural scaling factor. Note that this is interesting only when the answer to the whole problem is at least $\Omega(n^r)$. This holds for “dense” problems where there are $\Omega(n^r)$ clauses.] The question arises: what is the best $\text{poly}(1/\epsilon)$ in this result? Alon, de la Vega, Karpinski and Kannan [ADKK02] prove $O^*(1/\epsilon^4)$ suffices.

Theorem 3. [ADKK02] *Suppose $F(x_1, x_2, \dots, x_n)$ is a MAX-r-CSP formula. If Q is u.a.r. subset of $q = O^*(1/\epsilon^4)$ of the n variables, then, for F^Q , the induced formula on Q , we have with probability at least 99/100,*

$$\left| \text{MAX}(F) - \frac{n^r}{q^r} \text{MAX}(F^Q) \right| \leq \epsilon n^r.$$

There are two parts to the theorem: first asserts that $\frac{n^r}{q^r} \text{MAX}(F^Q) \geq \text{MAX}(F) - \epsilon n^r$. This is simple: if one just takes the truth assignment to $\{x_1, x_2, \dots, x_n\}$ which attains $\text{MAX}(F)$, then usual facts about sampling can be used to prove that the SAME assignment to the sampled variables satisfies (whp) at least $\frac{q^r}{n^r} \text{MAX}(F) - \epsilon q^r$ of the clauses of F^Q . The other part is the non-trivial one - the reason is we have to rule out ANY assignment to the sampled variables from satisfying too many clauses. Indeed, this raises a basic question:

When can we say for a maximization problem that a sampled induced subproblem gives a good estimate of the answer to the whole problem ?

The non-trivial part is: show that for small problem (the induced one on the sample), no solution gives an unduly high value. [Traditional sampling arguments tackle the other part easily.] A situation with a simple answer is bounded Linear Programming: it is easy to see that if a system of linear inequalities $Ax \leq b; 0 \leq x_i \leq 1$ in n variables has a solution, then, for a random subset Q of the n variables, the induced problem (slightly relaxed)

has a solution too: $A^Q x^Q \leq \frac{q}{n}b + \delta$; $0 \leq x_i \leq 1$ for $i \in Q$ where A^Q consists of the columns of A corresponding to the Q variables. But the converse is also true here using LP duality: if $Ax \leq b$ has no solution with $0 \leq x_i \leq 1$, then duality tells us that there is one combination of the inequalities which has no solution; this is equivalent to the existence of a $u \geq 0$ such that $\sum_j (u^T A)_j^- > u^T b$. Now for this u , we can show by traditional sampling that we have: $\sum_{j \in Q} (u^T A)_j^- > \frac{q}{n}(u^T b) - \delta$ demonstrating that there is no solution to a slight tightening of the sampled LP. This simple result for LP is used as part of the proof of Theorem 3.

Another result of a similar flavor about induced subproblems also goes into the proof of Theorem 3 and is worth mentioning independently here. Suppose A is a large $n \times n$ (note: it is square) matrix. If we pick a random subset Q of $[n]$ and look at the induced submatrix A^Q of A on $Q \times Q$, how does the cut norm of A^Q relate to the cut norm of A ? It is easy to see that $\|A^Q\|_{\square} \geq \frac{q^2}{n^2} \|A\|_{\square} - \delta$, where δ is small, since, we could take the subsets S, T of $[n]$ which maximize $|A(S, T)|$ and argue by traditional Statistics that $|A(S \cap Q, T \cap Q)| \geq \frac{q^2}{n^2} |A(S, T)| - \delta$. The theorem below by Rudelson and Vershynin asserts a converse which is harder to prove. It is an improvement of a theorem in [ADKK02] and is proved using some Functional Analysis techniques.

Theorem 4. [RV07] *Let $\epsilon > 0$ and suppose A is an $n \times n$ matrix with $\|A\|_F \in O(n)$; $\|A\|_{\square} \in O(\epsilon n^2)$; $|A_{ij}| \leq O(1/\epsilon)$. Then if Q is a u.a.r. subset of $\{1, 2, \dots, n\}$ with $|Q| = q \in \Omega(1/\epsilon^2)$ and A^Q is the $q \times q$ submatrix of A with entries from $Q \times Q$, then*

$$E\|A^Q\|_{\square} = O(\epsilon q^2).$$

Open Question [ADKK02] actually proved such a result for r -tensors for any fixed r ; but their proof required $q \in \Omega^*(1/\epsilon^4)$. Does a result as above with $O(1/\epsilon^2)$ hold for r -tensors? The issue is that the techniques from Functional Analysis are no more available for $r > 2$. (cf. also the next open question on approximating the cut norm has this flavor.)

It is not difficult to show that the problem of finding the cut-norm is MAX-SNP hard by a reduction from Max-Cut. Interestingly, using a deep result from Mathematics called Grothendik inequality, Alon and Naor [AN06] were able to show:

Theorem 5. [AN06] *The cut norm of matrices can be approximated to within a factor of 1.782 in polynomial time.*

Their approach is: the cut norm problem can be reduced to the following problem:

$$\text{MAX} \sum_{i,j} A_{ij} x_i y_j \text{ subject to } x_i, y_j \in \{-1, +1\}.$$

This Integer Program has a standard Semi-Definite Programming relaxation, where the ± 1 variables x_i, y_j are replaced by vector variables u_i, v_j , required to be of length 1:

$$\text{MAX} \sum_{i,j} A_{ij} (u_i \cdot v_j) \text{ subject to } |u_i| = |v_j| = 1.$$

The theorem of Grothendik proves that the optimal value of the SDP is at most a factor of 1.782 times the optimal value of the integer program. This automatically yields a constant factor approximation to the value of the integer program. But, finding a rounding procedure to achieve this was both non-trivial and first developed in [AN06].

Open Problem Develop $O(1)$ factor for cut norms of r -tensors. Note that the natural Semi Definite Program for $r = 2$ does not extend to $r = 3$.

In the last section, we saw the weak-regularity Lemma and a notion of distance between two graphs. Borgs, Chayes, Lovász, Sós, Szegedy and Veszteg [BCL⁺06] defined other interesting notions of graph distances and graph limits. They generalize the notion of graph distances to graphs with different numbers of vertices. While the definition in Lemma 2 viewed the vertex sets of the two graphs having a fixed 1-1 mapping (labeled vertices), this is no more possible and relabeling of each vertex set as well as mapping one to the other have to be allowed. We do not give the precise definitions here. Suppose we do the partition of the vertex set as in Lemma 2. Then we could represent each V_r by a compound vertex and put an edge between compound vertices r, s of weight equal to $(\text{number of edges between } V_r \text{ and } V_s) / |V_r| |V_s|$. Then the Lemma is really saying that the compressed graph and the original are close in some metric. [This intuition needs to be formalized into a definition of distance between graphs.] They then use Theorem 3 to show:

Theorem 6. [BCL⁺06] *Let G be a simple graph and $\epsilon, \delta > 0$. Then the induced sub-graph of G on a random subset of $2^{1/\delta\epsilon^2}$ nodes is ϵ close to G with probability at least $1 - \delta$.*

They use this theorem in their extensive work on Graph Limits. Their notions facilitate the understanding of very large graphs which can be viewed

as limits; but also can be approximated in the above sense by smaller graphs (with something akin to “compound vertices”.) This raises the following somewhat loosely phrased:

Open Question Can we define a notion of limits for matrices and more generally r -tensors and apply these to derive theorems similar to [BCL⁺06]?

4 Non-Uniform Sampling

Clearly, uniform sampling of rows/columns will not solve all problems. Indeed, if we have a matrix with just one non-zero row and all other rows were just 0's and we draw uniform sample of rows, we are likely to see only zeros and miss the all-important row. Less trivial examples are when only a small number of rows contain significantly higher absolute value entries than others. From the last sections, we can show that u.a.r. samples yield an approximation B to the given matrix A with error in cut norm of at most ϵmnM , where M is the max absolute value of an entry of A . It can also be shown that with $\text{poly}(1/\epsilon)$ u.a.r. samples, we can make $\|A - B\|_2 \leq \epsilon \sqrt{mn}M$, the point being (briefly) $\|A - B\|_{\square} = x^T(A - B)y$, where $|x| = \sqrt{m}$ and $|y| = \sqrt{n}$, so the $\|A - B\|_2$ error is $1/\sqrt{mn}$ times the $\|A - B\|_{\square}$ error. But this amount of error is not suitable for many applications.

A more useful error bound is given in Lemma (4), for not only matrices, but also tensors. Completely analogous to the matrix case, we make the following definitions: For an r -tensor A , and r vectors w, x, y, z, \dots , $A(w, x, y, z, \dots)$ is defined as $\sum_{i,j,k,l,\dots} A_{i,j,k,l,\dots} w_i x_j y_k z_l \dots$. [It is analogous to the quadratic form $x^T A y = \sum_{i,j} A_{ij} x_i y_j$ for matrices.] The Frobenius norm of A , denoted $\|A\|_F$ is again the square root of the sum of squares of the entries. The “spectral norm” of A denoted $\|A\|_2$ is the maximum over all unit length vectors w, x, y, z, \dots of $A(w, x, y, z, \dots)$. A rank-1 r -tensor is the outer product of r vectors, denoted $w \otimes x \otimes y \otimes z \dots$ whose i, j, k, l, \dots th entry is $w_i x_j y_k z_l \dots$. We say that a tensor has rank at most k if it can be expressed as the sum of k rank-1 tensors.

Lemma 4. [dlVKKV05] *For any A , $\epsilon > 0$, there exist a tensor B of rank at most $1/\epsilon^2$ such that*

$$\|A - B\|_2 \leq \epsilon \|A\|_F. \quad (1)$$

The simple proof of the Lemma will be given shortly. A polynomial time sampling based algorithm is also available for producing such an approximation, but the algorithm given by de la Vega, Karpinski, Kannan and Vempala [dlVKKV05] is non-trivial.

Theorem 7. [dlVKKV05] *For any $A, \epsilon > 0$, we can find a tensor B of rank at most $4/\epsilon^2$ in time $(n/\epsilon)^{O(1/\epsilon^4)}$ such that with probability at least $3/4$ we have*

$$\|A - B\|_2 \leq \epsilon \|A\|_F.$$

For matrices, traditional singular value decomposition gives us a polynomial time algorithm, but, we will see a sampling-based algorithm which in essence can be made constant time after 2 passes through the matrix. In the case of tensors, no previous polynomial time algorithm was known at all. Before giving the proofs/algorithms, we will motivate the error bound of $\|A - B\|_2 \leq \epsilon \|A\|_F$ of (1) by three application areas.

The first motivating area is Principal Component Analysis (PCA). Here, one often assumes that the top “few” singular values dominate. (In fact, that is in the first place one of the two justifications for making a low rank approximation. The other possible motivation for making a low-rank approximation is “de-noising” - where one assumes that the top few singular value components are the real data and the others are possibly noise- for example in Latent Semantic Indexing [DFLD88].)

PCA Assumption: The data consists of an $m \times n$ matrix A . The top k singular values $\sigma_1, \sigma_2, \dots, \sigma_k$ contain $1 - \epsilon$ of the “spectrum”, where $k \ll m, n$. More precisely,

$$[\textbf{Strong-PCA}] \sigma_1^2 + \sigma_2^2 + \dots + \sigma_k^2 \geq (1 - \epsilon) \|A\|_F^2.$$

We need only a weaker version of this:

$$[\textbf{Weak-PCA}] \sigma_1^2 + \sigma_2^2 + \dots + \sigma_k^2 \geq \Omega(\|A\|_F^2).$$

Under this assumption, (1) translates to a “relative error ϵ ”.

A second area is Discrete Optimization. As we saw, the max-cut problem can be solved to additive error $\epsilon n^2 M$ for n - node graphs where the edge weights are all at most M . This however is relative error ϵ only in case the total of all edge weights is $\Omega(n^2 M)$; if $M \leq 1$, this requires the graph to be dense. This raises the question:

Can we solve non-dense max cut problem to *relative error* ϵ ? In general, these problems are NP-hard. But an important special case, it turns out can be solved in polynomial time - namely when the edge weights satisfy the triangle inequality, as was shown using other methods[dLVK01]. A unified polynomial time algorithm using Theorem 7 for this problem and other weighted versions of MAX-2-CSP problems is developed in [dLVKKV05]. [In fact, they do this with a weaker condition than triangle inequality for all MAX-2-CSP problems.]

A third area is tensors. Many algorithms are known and used in practice for finding low-rank approximations to tensors [Kru89]. But as remarked earlier, neither the theory nor the algorithms are anywhere as nice as for matrices. There are solid reasons - NP-hardness [Has90], [HhL09] and non-uniqueness/existence. But beyond all this, is a basic question - what is it that we can find provably in polynomial time ? Theorem (7) seems to be a first step. The algorithm for Theorem (7) (which we will outline soon) is quite different from other known heuristics and draws on new uses of sampling in a vein somewhat similar to the maximum rectangle problem. Also, it turns out that the error bound in (1) suffices to tackle MAX-r-CSP problems where the weights satisfy a natural generalization of the triangle inequality to higher dimensions. Unweighted dense MAX-r-CSP's are a special case of this.

Proof of Lemma (4): If $\|A\|_2 \leq \epsilon \|A\|_F$, then we are done. If not, there are w, x, y, z, \dots , all of length 1 such that $A(w, x, y, z, \dots) \geq \epsilon \|A\|_F$. Now consider the r -dimensional array

$$B = A - (A(w, x, y, z, \dots)w \otimes x \otimes y \otimes z \dots \dots \dots)$$

[This is of course basically a rank-1 update.] It is easy to see that $\|B\|_F^2 = \|A\|_F^2 - (A(w, x, y, z, \dots))^2$. We may repeat on B and clearly this process will only go on for at most $1/\epsilon^2$ steps.

From the proof of the lemma, it is clear that again, the basic algorithmic question is to find a w, x, y, z, \dots all of unit length, maximizing $A(w, x, y, z, \dots)$ to within additive error $\epsilon \|A\|_F$. We will present the algorithm for tensors later. First, we will tackle matrices by sampling.

5 Sampling in large matrices

Numerical Analysis gives us sophisticated polynomial time algorithms for many matrix problems to do with spectral analysis. Here, the focus is on

using sampling to solve very large matrix problems approximately. First, we look at matrix multiplication. The product of two matrices A, B can be written as

$$AB = \sum_i A_i B^i,$$

where A_i (B^i respectively) is the i column of A (row of B , respectively). An immediate thought is to estimate the sum from a random sample of i 's. Consider a random sample of s i 's picked in i.i.d. trials. Let p_1, p_2, \dots, p_n be the probabilities of picking $1, 2, \dots, n$ respectively in each trial. If i_1, i_2, \dots, i_s are the samples, then

$$X = \frac{1}{s} \sum_{t=1}^s \frac{1}{p_{i_t}} A_{i_t} B^{i_t}$$

is easily seen to be an unbiased estimator of AB . [I.e., $EX = AB$ entry-wise.] We would like to measure the variance, but this quantity depends on which entry we are talking about. Here, we make a simple-minded, important decision - let's look at the sum of variances of all the entries of X . This quantity, which we denote $\mathbf{Var}X$ is seen to satisfy:

$$\mathbf{Var}X \leq \frac{1}{s} \sum_{i=1}^n \frac{1}{p_i} |A_i|^2 |B^i|^2.$$

A case of much interest is when $B = A^T$, when this simplifies to

$$\frac{1}{s} \sum_{i=1}^n \frac{1}{p_i} |A_i|^4.$$

By Calculus, one can see that this is minimized when the p_i are proportional to $|A_i|^2$. [Indeed it is not hard to show that these p_i are the minimizer of the actual variance, not just the upper bound here.] This leads to the following probability distribution for sampling the columns of a matrix which turns out to have many nice properties:

Length Squared Sampling : Pick a column with probability proportional to sum of squares of its entries.

Length squared sampling was first introduced by Frieze, Kannan and Vempala [FKV98]. Its applications to clustering were studied by Drineas, Frieze, Kannan, Vempala and Vinay [DFK⁺04]. The application to matrix multiplication is from [DK01]. See also [DKM06a].

[FKV98] proves that if we draw a sample of columns according to the length squared distribution and do an SVD on the sampled columns, this gives an low-rank approximation to A with provable error bounds. We state this below (without proof).

Algorithm: Fast-SVD

1. Sample s columns of A from the squared length distribution to form a matrix C .
2. Find $u^{(1)}, \dots, u^{(k)}$, the top k left singular vectors of C .
3. Output $\sum_{t=1}^k u^{(t)} u^{(t)T} A$ as a rank- k approximation to A .

The matrix $\sum_{t=1}^k u^{(t)} u^{(t)T} A$ is really just the “projection” of A on the space spanned by the $u^{(t)}$ and so the theorem below says that A projected to the top singular space of C (instead of the usual singular space of A) is a good low-rank approximation to A . [A_k is the best rank k approximation given by SVD.]

Theorem 8. [FKV98],[FKV04] *The rank- k matrix found by Algorithm Fast-SVD (call it \tilde{A}) satisfies:*

$$\mathbf{E} \left(\|A - \tilde{A}\|_F^2 \right) \leq \|A - A_k\|_F^2 + 2\sqrt{\frac{k}{s}} \|A\|_F^2$$

$$\mathbf{E} \left(\|A - \tilde{A}\|_2^2 \right) \leq \|A - A_k\|_2^2 + \frac{2}{\sqrt{s}} \|A\|_F^2.$$

In fact the kind of error bound in the theorem is optimal in terms of the number of rows sampled; this was shown in [BY03].

[FKV98] and [FKV04] in fact apply the sampling once more - to pick a sample of rows of C according to the length-squared distribution. Then, it turns out that fining the SVD of the constant-sized matrix (with the sampled

rows of C) suffices to give us a low-rank approximation to A . But the proof of this is more complicated. The reason is that from the sampled rows of C , one gets the right singular vectors of C , but only approximately. The error turns out to be bounded by $\epsilon\|C\|_F$. [It would be better if the error bound was relative, in terms of the singular values themselves. But length-squared sampling does not give this.] Then for the “low” singular values of C (less than $\epsilon\|C\|_F$), the approximation is no good. So, one has to throw out these low ones (these are in a sense “near-singularities”) See [DKM06b] for a detailed explanation of the method and some improvements.

An improvement of the error bound, still using length-squared sampling was achieved using sophisticated techniques from the field of Probability in Banach spaces by Rudelson and Vershynin. Their result stated below picks a sample of s rows from A , where s is almost linear in a quantity r , they call the numerical rank of A ; $r = \|A\|_F^2/\|A\|_2^2$. [Recall the PCA assumptions; under even the weak PCA assumption, r is $O(1)$.] Their error bound is also better in that it involves $\|A\|_2$, rather than $\|A\|_F$.

Theorem 9. [RV07] *Suppose A is an $m \times n$ matrix with numerical rank $r = \|A\|_F^2/\|A\|_2^2$. Let $\epsilon, \delta \in (0, 1)$ and $s \in \Omega^*(r/\epsilon^4\delta)$. Let B be a set of s rows of A picked in s i.i.d. trials, each according to length-squared and let $u^{(1)}, \dots, u^{(k)}$ be the top k right singular vectors of B . Then, $\tilde{A} = A \sum_{t=1}^k u^{(t)} u^{(t)T}$ satisfies the following with probability at least $1 - 2e^{-c/\delta}$:*

$$\|A - \tilde{A}\|_2 \leq \sigma_{k+1}(A) + \epsilon\|A\|_2.$$

Note also that this is a high probability (with exponential tails) rather than just in expectation.

In another application of length-squared sampling, [SV09] shows that if we run an iterative equation solver for an overdetermined system of equations with a Kaczmarz iteration (where one uses a violated equation to modify the current solution) with the added twist that the violated equation is picked according to the length squared distribution, then one gets a guaranteed rate of convergence; the reader is referred to the paper for details.

Length-squared sampling can also be used for tensors and is the basic ingredient in the proof of theorem 7; recall that we needed an algorithm to find for a tensor A , the maximum value of $A(w, x, y, z, \dots)$ to within $\epsilon\|A\|_F$. The idea behind the algorithm for this is to imitate the steps of the algorithm for the maximum rectangle problem:

1. If we knew the optimizing x, y, z, \dots , then the optimizing w is easy to find: it is just the vector $A(\cdot, x, y, z, \dots)$ (whose i th component is $A(e_i, x, y, z, \dots)$) scaled to length 1.
2. Now, $A(e_i, x, y, z, \dots) = \sum_{j,k,l,\dots} A_{i,j,k,l,\dots} x_j y_k z_l \dots$. The sum can be estimated by having just a few terms. But, an important question is: how do we make sure the variance is not too high, since the entries can have disparate values ?
3. Length squared sampling works ! [Stated here without proof.]

Achlioptas and McSherry [AM07] developed a different randomized algorithm for low-rank approximations of matrices - they sample individual entries independently and show using Random Matrix theory that with those on hand, we can get a good approximation to the matrix in spectral norm. Their results also have a bearing on “Compressed Sensing” in that they are able to infer something about the whole matrix from a random sample of entries.

6 CUR: An interpolative low-rank approximation

We found in the last section an implicit low-rank approximation to A ; implicit because, the actual approximation needed us to multiply A by the vectors $u^{(t)}$. In this section, we wish to describe an algorithm to get an explicit approximation of any matrix A given just a sample of rows and a sample of columns of A . Clearly if the sample is picked according to the uniform distribution, this attempt would fail in general. We will see that again the length squared distribution comes to our rescue; indeed, we will show that if the samples are picked according to the length squared or approximate length squared distributions, we can get an approximation for A . Again, this will hold for an arbitrary matrix A .

First suppose A is a $m \times n$ matrix and R (R for rows) is a $s \times n$ matrix constructed by picking s rows of A in i.i.d. samples, each according to approximate length-squared distribution. Similarly, let C (for columns) be a $m \times s$ matrix consisting of columns picked according to the length squared distribution on the columns. The motivating question for this section is:

Can we get an approximation to A given just C, R ? An affirmative answer is given in the theorem below first proved by Drineas and Kannan. Here, one does not need the sampling probabilities to be exactly proportional to length squared; it suffices to have the probability of drawing column i to be at least its length squared / $(c \cdot \|A\|_F^2)$, where c is a constant. We call this approximate length squared sampling.

Theorem 10. [DK03], [DKM06c] Suppose C (respectively R) consists of a sample of $s \geq \Omega^*(k/\epsilon^4)$ columns (respectively rows) of A drawn in s i.i.d. trials, each according to (approximate) length squared probabilities. Then, from C and R , we can find a $s \times s$ matrix U so that

$$\begin{aligned} E\|A - CUR\|_F &\leq \|A - A_k\|_F + \epsilon\|A\|_F \\ E\|A - CUR\|_2 &\leq \|A - A_k\|_2 + \epsilon\|A\|_F. \end{aligned}$$

Open Problem Improve the dependence of $1/\epsilon^4$ in the theorem.

The approximation of A by the product CUR is reminiscent of the usual PCA approximation based on taking the leading k terms of the SVD decomposition. There, instead of C, R , we would have orthonormal matrices consisting of the leading singular vectors and instead of U , the diagonal matrix of singular values. The PCA decomposition of course gives the best rank- k approximation, whereas what the Theorem shows for CUR is only that its error is bounded in terms of the best error we can achieve. There are two main advantages of CUR over PCA:

1. CUR can be computed faster from A and also we only need to make two passes over A which can be assumed to be stored on external memory.
2. CUR preserves the sparsity of A - namely C, R are columns and rows of A itself. (U is a small matrix since typically s is much smaller than m, n). So any further matrix vector products Ax can be approximately computed as $C(U(Rx))$ quickly.
3. It is an interpolative approximation : unlike SVD, where the singular vectors are linear combinations of A 's columns/rows, here C, R are actual columns/rows of A . An application illustrates the point. In doing, say, PCA on a ‘‘patient-gene’’ matrix in a Biological application (with entry i, j giving the gene expression of gene j for patient i), one gets a result which says that ‘‘these few linear combinations of genes/patients

are important in explaining the data”, where the linear combinations may involve negative weights as well as positive ones. Instead in CUR, we get a collection of individual genes/patients explaining the data, arguably providing better intuition. See for example [PMJ⁺07] for this kind of application.

The CUR approximation has been extended to tensors as well by Mahoney, Maggioni and Drinesa [MMD]. There are many improvements and applications of CUR - see [MD09]. An important modification is given by Sun Xi, Zhang and Faloutsos [SXZF07]. They make the observation that the length squared sampling used in the original CUR algorithm may result in a lot of duplicates. They remove duplicates and reweight the unique columns/rows remaining by the square root of the number of copies. They show that with this new reweighting, one gets good approximations. More importantly, they have done empirical studies with Datamining applications to show the effectiveness in terms of space and time with the new approximation.

We also briefly describe an application of CUR to Recommendation Systems by Drineas, Kerneidis and Raghavan [DKR02]. The central object in a Recommendation System is the customer-preference matrix whose (i, j) th entry is the preference of customer i for product j . The basic question is: given a small sample of entries from the matrix (collected customer-product data) how does one make good recommendations to other customers. The idea of using CUR in [DKR02] is a departure from what we have discussed so far. So far, we had (somewhere) the whole matrix and used sampling to infer its properties, it being too large to deal with in full. Here, even collecting the matrix is expensive; we wish to infer its missing entries from the sample we know. But note that the techniques of this survey do apply to this “reverse engineering” problem. Indeed, [DKR02] prove that under some assumptions, we can make good recommendations with just a handful of customers’ complete preferences (some complete rows) and all customer preferences for a handful of products (columns). Much care has to be taken, since the sampling cannot be assumed to be according to Length-squared.

The above raises a more general question. Can we bring costs of accurate data collection into our measures of efficiency ? We loosely formulate a candidate problem on these lines:

Open Question Suppose we wish to solve a large Linear Program : $\max c \cdot x$ subject to $Ax \leq b$. But suppose we have to collect each piece

of data - each A_{ij} costs c/ϵ^2 to get with accuracy $\pm\epsilon$. Find an efficient method for data collection + solution to within error $\pm\delta$, where the cost is the weighted sum of the data collection cost plus the running time.

7 Relative Error Approximation

If A_k is the best rank k approximation (from SVD), then a natural way to measure error of any rank k approximation is relative to the “residue” of the spectrum, namely relative to $\|A - A_k\|_F$. So, we say a rank k approximation B makes relative error ϵ if

$$\|A - B\|_F \leq (1 + \epsilon)\|A - A_k\|_F.$$

From simple examples, it is easy to see that length-squared sampling does not do this in general.

For multiplicative $(1 + \epsilon)$ -approximation, Har-Peled [HP05] gave a linear time algorithm that requires $O(\log n)$ passes over the input matrix. Deshpande and Vempala [DV06] improved this to $O(k)$ passes using volume sampling. Both these algorithms use adaptive sampling of [DRVW06] as a subroutine. Drineas, Mahoney and Muthukrishnan [DMM06] gave a different algorithm, where the sampling probabilities are computed using SVD, that achieves the same approximation ratio but takes more time because of the initial computation of SVD. Finally, Sarlös [Sar06] gave a linear time 2-pass algorithm for multiplicative $(1 + \epsilon)$ -approximation that uses a small number of linear combinations of all the rows instead of a subsample; his algorithm which we call isotropic random projection is described below.

First, volume sampling is a generalization of length-squared sampling. We pick subsets of k rows instead picking rows one by one. The probability that we pick a subset S is proportional to the volume of the k -simplex $\Delta(S)$ spanned by these k rows along with the origin. The raw method will give us a factor $(k + 1)$ approximation (in expectation). Incidentally, it also proves that any matrix has k rows whose span contains a such an approximation. Moreover, this bound is tight, i.e., there exist matrices for which no k rows can give a better approximation.

Lemma 5. [DV06] *Let S be a random subset of k rows of a given matrix A chosen with probability*

$$P_S = \frac{\text{Vol}(\Delta(S))^2}{\sum_{T:|T|=k} \text{Vol}(\Delta(T))^2}.$$

Let \tilde{A} be the projection of A to the span of S and let \tilde{A}_k be the best rank k approximation to \tilde{A} . Then,

$$\mathbf{E} (\|A - \tilde{A}_k\|_F^2) \leq (k+1)\|A - A_k\|_F^2.$$

More work is needed to convert this to a relative error ϵ approximation (which we do not describe here.)

Isotropic random projection also gives relative error approximations to the optimal rank- k matrix with roughly the same time complexity. Moreover, it makes only *two* passes over the input data.

The idea behind the algorithm can be understood by going back to the matrix multiplication algorithm described in section 5. There to multiply two matrices A, B , we picked random columns of A and rows of B and thus derived an estimate for AB from these samples. The error bound derived was additive and this is unavoidable. Suppose that we first project the rows of A randomly to a low-dimensional subspace, i.e., compute AR where R is random and $n \times k$, and similarly project the columns of B , then we can use the estimate ARR^TB . For low-rank approximation, the idea extends naturally: first project the rows of A using a random matrix R , then project A to the span of the columns of AR (which is low dimensional), and finally find the best rank k approximation of this projection. The algorithm is:

1. Let $l = Ck/\epsilon$ and R be a random $n \times l$ matrix; compute $B = AR$.
2. Project A to the span of the columns of B to get \tilde{A} .
3. Output \tilde{A}_k , the best rank- k approximation to \tilde{A} .

Theorem 11. [Sar06] *Let A be an $m \times n$ real matrix with M nonzeros. Let $0 < \epsilon < 1$ and R be an $n \times l$ random matrix with i.i.d. Bernoulli entries with mean zero and $l \geq Ck/\epsilon$ where C is a universal constant. Then with probability at least $3/4$,*

$$\|A - \tilde{A}_k\|_F \leq (1 + \epsilon)\|A - A_k\|_F$$

and \tilde{A}_k can be computed in two passes over the data in $O(Ml + (m+n)l^2)$ time using $O((m+n)r^2)$ space.

8 Applications to Clustering, Mixtures

Spectral methods are widely used for clustering and partitioning problems. Not many worst-case results have been proved; there are exceptions for special classes of graphs like planar graphs [ST07]. In general, spectral methods have been proven to work correctly with high probability under some assumptions on the generative model of the data. One class of such problems is the Planted Problems, where we are given a random graph modified by a “planted part” and the objective is to find the planted part. We describe an instance of this.

Consider a graph G which is the union of a purely random graph $G_{n,1/2}$ and an unknown clique on vertex set P , where $p = |P|$ is given. The problem is to recover P . If $p \geq c(n \log n)^{1/2}$ then with high probability, it is easy to recover P as the p vertices of largest degree. Alon, Krivelevich and Sudakov [AKS98], using spectral analysis, were able to improve this to $p = \Omega(n^{1/2})$.

Let A_G denote the adjacency matrix of G . The spectral approach of [AKS98] essentially maximizes $x^T A x$ over vectors x with $|x| = 1$, expecting that the optimal solution is close to u , defined by $u_i = p^{-1/2} 1_{i \in P}$, (u is the scaled characteristic vector of P) so that we may recover P from the optimal solution.

Frieze and Kannan [FK08] define a natural 3-dimensional array A related to the given graph : A_{ijk} will be ± 1 depending on whether the parity of the number of edges among the vertices i, j, k is odd or even respectively. They show that as long as $p \in O(n^{1/3}(\log n)^4)$, the maximum of the cubic form $\sum_{i,j,k} A_{ijk} x_i x_j x_k$ as the vector x varies over the unit ball is attained close to u , so that if we can find this maximum, then we can recover the clique. However, unlike the case of the quadratic form, where the maximization was an eigenvalue computation which is well-known to be doable in polynomial time, there are in general no known polynomial time algorithms for maximizing cubic forms. So, the existential result does not automatically lead to an algorithm and this is left as an open question.

Open Question Suppose a $n \times n \times n$ array A is constructed as above from $G_{n,1/2}$ plus a planted clique of size $p \in \Omega(n^{1/3}(\log n)^c)$. Then can we maximize the function $\sum_{i,j,k} A_{ijk} x_i x_j x_k$, $|x| \leq 1$, even within $O(1)$ factors in polynomial time ?

Brubaker and Vempala [BV09] have generalized this to r tensors, where they show that maximizing over an r tensor whose entries are the parity of the number of edges in r cliques can find hidden cliques of size $n^{1/r}$.

The computational question of approximately maximizing the r -ary forms is open.

Another class of models is in a sense also planted - there is a hidden partition which dictates probabilities. For example, McSherry [McS01] (following earlier papers) considers a model in which n objects are divided into k clusters ($k \ll n$) T_1, T_2, \dots, T_k . There is a number $p_{rs} \in [0, 1]$ which is the probability of each edge between a vertex in T_r and one in T_s . Edges are chosen independently and we are given the resulting random graph on n vertices. Our job is to find the partition and p_{rs} of the generating model. This can be summarized as: we are given a 0-1 matrix A and are to find EA , where the expectation is entry-wise. [McS01] shows that under some technical conditions, spectral methods will yield the answer. Here is a quick idea of the method and its use of the deep results from the theory of random matrices. The matrix $A - \mathbf{E} A$ has random independent entries each with mean 0. The following celebrated theorem was first stated qualitatively by the physicist Wigner and proved by Füredi and Komlos [FK81]. See also [Vu05].

Theorem 12. *Suppose A is a symmetric random matrix with independent (above-diagonal) entries each with standard deviation at most ν and bounded in absolute value by 1. Then, with high probability, the largest eigenvalue of $A - \mathbf{E} A$ is at most $c\nu\sqrt{n}$.*

The strength of this Theorem is seen from the fact that each row of $A - \mathbf{E} A$ is of length $O(\nu\sqrt{n})$, so the Theorem asserts that the top eigenvalue amounts only to the length of a constant number of rows; i.e., there is almost no correlation among the rows (since the top eigenvalue $= \max_{|x|=1} \|(A - \mathbf{E} A)x\|$ and hence the higher the correlation of the rows in some direction x , the higher its value). Thus one gets whp an upper bound on the spectral norm of $A - EA$:

$$\|A - \mathbf{E} A\| \leq c\nu\sqrt{n}.$$

Now, we can (approximately) find EA by doing SVD on A with the help of the following simple lemma.

Lemma 6. [AM07] *Suppose A, B are $m \times n$ matrices with $\text{rank}(B) = k$. If \hat{A} is the best rank k approximation to A , then*

$$\|\hat{A} - B\|_F^2 \leq 5k\|A - B\|^2.$$

While these ideas are clean, it turns out that they only help cluster “most” points correctly. The others are corrected in a messy “clean-up” phase. There has been progress on clustering in generative models: [AFKM01],[DHKS05],[DHKM07]. But the messiness of the clean-up phase haunts the field and raises the following:

Open Problem Clean up the clean-up phase of clustering algorithms for generative models (or dispense with it).

Another well-studied clustering problem has to do with learning mixtures of Gaussians and other probability densities. We only describe the part of this area which has to do with spectral algorithms. A provable connection to spectral method was struck by Vempala and Wang [VW04]. They proved an elegant result that given samples from a mixture of k spherical Gaussians, the k -dimensional SVD subspace of the matrix whose rows are the samples contains all the centers. With the space of centers in hand, one can project to that subspace and learn in it. Extensions of this were given in [KSV08] and [AM05]. Two interesting variants of PCA have been proposed recently by Brubaker and Vempala - isotropic PCA [BV08] and robust PCA [Bru09] which tackle Gaussian mixture learning problems not amenable to standard PCA.

While we do not go into the subject of Spectral Partitioning of graphs, we briefly mention that there are many ways to partition nodes given edge weights which are to be treated as pairwise similarities between vertices. A well-used method is to normalize first each row sum to be 1, then find the second largest eigenvalue and corresponding eigenvector of the stochastic matrix. Then we partition the vertex set into 2 subsets: those with coordinate in the second eigenvector and those with low coordinates. The cut-off can be chosen. This algorithm and its variations are widely used [SM00]. Not many proofs of error bounds are known though. [ST07] prove bounds for planar graphs. [KVV04] prove that if one repeats this partitioning procedure on the subgraphs, then we can ensure that the graph is ultimately split into parts of high conductance with not too much edge weight “wasted” between different parts. [KM08] prove better bounds for planar graphs.

Acknowledgements I am grateful to Alan Frieze and Santosh Vempala for their collaboration on work reported here and to Santosh also for his comments on the manuscript. Thanks to all my coauthors.

References

- [ADKK02] N. Alon, W.F. DeLaVega, R. Kannan, and M. Karpinski, *Random sub-problems of max-snp problems*, 34th STOC (2002), 668–677.
- [AE02] G. Andersson and L. Engebretsen, *Property testers for dense constraint satisfaction programs on finite domains*, Random Structures Algorithms **21** (2002), 1432.
- [AFKM01] Y. Azar, A. Fiat, A. Karlin, and F. McSherry, *Spectral analysis of data*, Proc. of STOC, 2001, pp. 619–626.
- [AKK95] S. Arora, D. Karger, and M. Karpinski, *Polynomial time approximation schemes for dense instances of np-hard problems*, 27th STOC (1995), 284–293.
- [AKS98] N. Alon, M. Krivelevich, and B. Sudakov, *Finding a large hidden clique in a random graph*, Random Structures and Algorithms **13** (1998), 457–466.
- [AM05] D. Achlioptas and F. McSherry, *On spectral learning of mixtures of distributions*, Proc. of COLT, 2005.
- [AM07] ———, *Fast computation of low-rank matrix approximations*, J. ACM **54** (2007), no. 2.
- [AN06] N. Alon and A. Naor, *Approximating the cut-norm via grothendieck’s inequality*, SIAM J. Comput. **35**(4) (2006), 787–803.
- [ARH⁺] N. Alon, R.A.Duke, H.Lefmann, V.Rödl, and R.Yuster, *The algorithmic aspects of the regularity lemma*, Journal of Algorithms **16**.
- [BCL⁺06] C. Borgs, J. Chayes, L. Lovász, V. Sós, B. Szegedy, and K. Vesztergombi, *Graph limits and parameter testing*, STOC, 2006.
- [BP98] S. Brin and L. Page, *The anatomy of a large-scale hypertextual web search engine*, Proc. 7th international conference on World Wide Web (WWW), 1998, p. 107117.

- [Bru09] S. C. Brubaker, *Robust pca and clustering on noisy mixtures*, Proc. of SODA, 2009.
- [BV09] C. Brubaker and S. Vempala, *Random Tensors and Planted Cliques*, Proc. of APPROX-RANDOM, 2009. 406-415.
- [BV08] S. C. Brubaker and S. Vempala, *Isotropic pca and affine-invariant clustering*, Building Bridges Between Mathematics and Computer Science (M. Grötschel and G. Katona, eds.), Bolyai Society Mathematical Studies, vol. 19, 2008.
- [BW09] N. Bansal and R. Williams, *Regularity lemmas and combinatorial algorithms*, Proc. IEEE 50 th FOCS, 2009, pp. 745–755.
- [BY03] Z. Bar-Yossef, *Sampling lower bounds via information theory*, Proc. of STOC, 2003, pp. 335–344.
- [DFK⁺04] P. Drineas, A. Frieze, R. Kannan, S. Vempala, and V. Vinay, *Clustering large graphs via the singular value decomposition*, Machine Learning **56** (2004), 9–33.
- [DFLD88] S.T. Dumais, G.W. Furnas, T.K. Landauer, and S. Deerwester, *Using latent semantic analysis to improve information retrieval*, Proc. of CHI, 1988, pp. 281–285.
- [DHKM07] A. Dasgupta, J. Hopcroft, R. Kannan, and P. Mitra, *Spectral clustering with limited independence*, Proc. of SODA, 2007, pp. 1036–1045.
- [DHKS05] A. Dasgupta, J. Hopcroft, J. Kleinberg, and M. Sandler, *On learning mixtures of heavy-tailed distributions*, FOCS, 2005.
- [DK01] P. Drineas and R. Kannan, *Fast monte-carlo algorithms for approximate matrix multiplication*, FOCS, 2001, pp. 452–459.
- [DK03] ———, *Pass efficient algorithms for approximating large matrices*, SODA '03, 2003, pp. 223–232.
- [DKM06a] P. Drineas, R. Kannan, and M. Mahoney, *Fast monte carlo algorithms for matrices i: Approximating matrix multiplication*, SIAM J. on Computing **36** (2006), 132–157.

- [DKM06b] ———, *Fast monte carlo algorithms for matrices ii: Computing a low-rank approximation to a matrix*, SIAM J. on Computing **36** (2006), 158–183.
- [DKM06c] ———, *Fast monte carlo algorithms for matrices iii: Computing a compressed approximate matrix decomposition*, SIAM J. on Computing **36** (2006), 184–206.
- [DKR02] P. Drineas, I. Kerenidis, and P. Raghavan, *Competitive Recommendation Systems*, Proceedings of the 34th Annual ACM Symposium on Theory of Computing (2002), 82–90.
- [dlV96] W. Fernandez de-la Vega, *MAX-CUT has a Randomized Approximation Scheme in Dense Graphs*, Random Structures and Algorithms **8** (1996), 187–199.
- [dlVK01] W. Fernandez de la Vega and C. Kenyon, *A randomized approximation scheme for metric max-cut*, J. Computer and System Sciences **63** (2001), 531–541.
- [dlVKKV05] W. Fernandez de la Vega, M. Karpinski, R. Kannan, and S. Vempala, *Tensor decomposition and approximation schemes for constraint satisfaction problems*, STOC '05, 2005, pp. 747–754.
- [DMM06] P. Drineas, M. W. Mahoney, and S. Muthukrishnan, *Subspace sampling and relative error matrix approximation: column-based methods*, Proc. of APPROX-RANDOM, 2006, pp. 316–326.
- [DRVW06] A. Deshpande, L. Rademacher, S. Vempala, and G. Wang, *Matrix approximation and projective clustering via volume sampling*, Theory of Computing **2** (2006), no. 1, 225–247.
- [DV06] A. Deshpande and S. Vempala, *Adaptive sampling and fast low-rank matrix approximation*, APPROX-RANDOM, 2006, pp. 292–303.
- [Fie75] M. Fiedler, *A property of eigenvectors of nonnegative symmetric matrices and its application to graph theory*, Czech. Math. J. **25** (1975), 619–637.

- [FK81] Z. Füredi and J. Komlós, *The eigenvalues of random symmetric matrices*, *Combinatorica* **1** (1981), no. 3, 233–241.
- [FK99] A. Frieze and R. Kannan, *Quick approximation to matrices and applications*, *Combinatorica* **19** (1999), no. 2, 175–200.
- [FK08] ———, *A new approach to the planted clique problem*, Proc. of FST & TCS, 2008.
- [FK999] ———, *A simple algorithm for constructing szemerédi’s regularity partition*, *Electronic journal of combinatorics* **6** (1999).
- [FKV98] A. Frieze, R. Kannan, and S. Vempala, *Fast monte-carlo algorithms for finding low-rank approximations*, Proc. of FOCS, 1998, pp. 370–378.
- [FKV04] A. Frieze, R. Kannan, and S. Vempala, *Fast monte-carlo algorithms for finding low-rank approximations*, *J. ACM* **51** (2004), no. 6, 1025–1041.
- [GGR98] O. Goldreich, S. Goldwasser, and D. Ron, *Property testing and its connection to learning and approximation*, *Journal of the ACM* **5** (1998), no. 4, 653–750.
- [GT08] B. Green and T. Tao, *The primes contain arbitrarily long arithmetic progressions*, *Annals of Mathematics* **167** (2008), 481547.
- [Has90] J. Hastad, *Tensor rank is np-complete*, *J. Algorithms* **11** (1990), 644–654.
- [HhL09] C. Hillar and Lek heng Lim, *Most tensor problems are np hard*, The Computing Research Repository (2009).
- [HP05] Har-Peled, *Low-rank matrix approximation in linear-time*, <http://valis.cs.uiuc.edu/~sariel/papers/05/lrank/>, 2005.
- [Kle99] J. Kleinberg, *Authoritative sources in a hyperlinked environment*, *Journal of the ACM* **46** (1999).
- [KM08] I. Koutis and G. L. Miller, *Graph partitioning into isolated, high conductance clusters: Theory, computation and applications to preconditioning*, Proc. of SPAA, 2008.

- [Kru89] J. B. Kruskal, *Rank, decomposition, and uniqueness for 3-way and n-way arrays*, Multiway Data Analysis, R. Coppi and S. Bolasco, eds. NorthHolland, Amsterdam, 1989, p. 718.
- [KSV08] R. Kannan, H. Salmasian, and S. Vempala, *The spectral method for general mixture models*, SIAM Journal on Computing **38** (2008), no. 3, 1141–1156.
- [KV08] R. Kannan and S. Vempala, *Spectral algorithms*, Foundations and Trends in Theoretical Computer science **4:3-4** (2008).
- [KVV04] R. Kannan, S. Vempala, and A. Vetta, *On clusterings: Good, bad and spectral*, J. ACM **51** (2004), no. 3, 497–515.
- [McS01] F. McSherry, *Spectral partitioning of random graphs*, FOCS, 2001, pp. 529–537.
- [MD09] M. W. Mahoney and P. Drineas, *Cur matrix decompositions for improved data analysis*, Proceedings of the National Academy of Sciences, vol. 106(3), 2009, pp. 697–702.
- [MMD] M. W. Mahoney, M. Maggioni, and P. Drineas, *Tensor-cur decompositions for tensor-based data*, SIAM Journal on Matrix Analysis and Applications **30(2)**.
- [PMJ⁺07] P. Paschou, M. W. Mahoney, A. Javed, J. Kidd, A. Pakstis, S. Gu, K. Kidd, and P. Drineas, *Intra- and inter-population genotype reconstruction from tagging snps*, Genome Research **17(1)** (2007), 96–107.
- [RV07] M. Rudelson and R. Vershynin, *Sampling from large matrices: An approach through geometric functional analysis*, Journal of the ACM **54** (2007), no. 4.
- [Sar06] T. Sarlós, *Improved approximation algorithms for large matrices via random projections*, FOCS, 2006, pp. 143–152.
- [SM00] J. Shi and J. Malik, *Normalized cuts and image segmentation*, IEEE Transactions on Pattern Analysis and Machine Intelligence **22** (2000), no. 8, 888–905.

- [ST07] D. A. Spielman and S. Teng, *Spectral partitioning works: Planar graphs and finite element meshes*, Linear Algebra and its Applications **421** (2007), no. 2-3, 284 – 305.
- [SV09] T. Strohmer and R. Vershynin, *A randomized kaczmarz algorithm with exponential convergence*, Journal of Fourier Analysis and Applications **15** (2009), 262–278.
- [SXZF07] J. Sun, Y. Xie, H. Zhang, and Christos Faloutsos, *Less is more: Compact matrix decomposition for large sparse graphs*, SIAM Conference on Data Mining, 2007, pp. 26–28.
- [TTV09] L. Trevisan, M. Tulsiani, and Salil Vadhan, *Regularity boosting and efficiently simulating every high-entropy distribution*, Proc. of 24th IEEE Conference on Computational Complexity, 2009.
- [Vu05] V. H. Vu, *Spectral norm of random matrices*, Proc. of STOC, 2005, pp. 423–430.
- [VW04] S. Vempala and G. Wang, *A spectral algorithm for learning mixtures of distributions*, Journal of Computer and System Sciences **68** (2004), no. 4, 841–860.